EmonTxV34CM_rfm69n.ino

This sketch should only be used when transmitting data to an emonPiCM that expects the radio data packet to use the “RFM69 Native” format. An emonPi or emonBase **not** using that format will be unable to receive the data from this sketch. (Use EmonTxV3CM.ino instead.)

This sketch uses the Continuous Monitoring library.

Setting up

The emonTx V3.4 has always been fitted with a RFM69CW radio module. Ensure the correct band for the module fitted is chosen, and an efficient aerial is present, if the output power is turned up beyond the default value of 25. The RFM69CW can be damaged beyond repair if it is operated at or near full power without the proper aerial.

If for any reason a RFM12B is fitted, that must be set in line 43 of the main sketch file:

#define RFM12B

The range of r.f. power levels, and the number of discrete levels available in the RFM12B is very limited – the nearest available power corresponding with the desired setting will be used.

The default format for data sent via the wired serial connection is suitable for the ESP8266 / emonhub EmonHubTx3eInterfacer.

To enable serial data in space-separated format, you must comment line 41 viz:

```
// #define EMONHUBTX3EINTERFACER
```

If a wired serial output is to be used exclusively, it is advisable to turn off the radio.

Data is transmitted, by radio or serially, approximately every 10 s. The “standard” Node IDs allocated to the emonTxV3.4 are 15 & 16.

Additional set-up options

At power-up, if you have a serial monitor connected via the FTDI connector and set to 115200 baud with “both NL & CR” selected, you will see a “Welcome” message giving the software version number, and a report of the configuration settings and calibration values.

```
emonTx V3.4 EmonLibCM Continuous Monitoring V0.00
OpenEnergyMonitor.org
No EEPROM config
Settings:
Group 210, Node 15, Band 433 MHz

Calibration:
vCal = 268.97
assumedV = 240.00
i1Cal = 90.90
i1Lead = 4.20
i2Cal = 90.90
i2Lead = 4.20
i3Cal = 90.90
```

```
i3Lead = 4.20
i4Cal = 16.67
i4Lead = 6.00
datalog = 9.85
pulses = 1
pulse period = 100
temp_enable = 1
Temperature Sensors found = 0 of 1
Temperature measurement is NOT enabled.

RF on
RFM69CW Freq: 433MHz Group: 210 Node: 15
```

You will then see the standard “readable” serial data output. It will be similar to this:

```
CT1 detected, i1Cal:90.90
CT2 detected, i2Cal:90.90
CT3 detected, i3Cal:90.90
CT4 detected, i4Cal:16.67
AC present
MSG:1,P1:590,P2:1,P3:0,P4:38,E1:1,E2:0,E3:0,E4:0,pulse:0
MSG:2,P1:585,P2:1,P3:0,P4:37,E1:3,E2:0,E3:0,E4:0,pulse:0

[etc]
```

If, however, `#define EMONHUBTX3EINTERFACER` is commented out, then the output will continue similar to this:

```
CT1 detected, i1Cal:90.90
CT2 detected, i2Cal:90.90
CT3 detected, i3Cal:90.90
CT4 detected, i4Cal:16.67
AC present
15 1 614 1 0 39 1 0 0 0 300.00 300.00 300.00 0
15 2 612 5 10 1 1 0 0 0 300.00 300.00 300.00 0

[etc]
```

At any time, you may enter the configuration and calibration mode by sending the characters **+++**[**Enter**]

If you successfully enter the access code, you will see an advisory and a menu of commands:

```
Entering Settings mode...

Available commands:
  l          - list the settings
  r          - restore sketch defaults
  s          - save settings to EEPROM
  v          - show firmware version
  z          - zero energy values
  x          - exit, lock and continue
  ?          - show this text again

  w<x>      - turn RFM Wireless data off: x = 0 or on: x = 1
  b<n>      - set r.f. band n = a single numeral: 4 = 433MHz, 8 = 868MHz,
                9 = 915MHz (may require hardware change)
```

```
  p<nn>     - set the r.f. power. nn - an integer 0 - 31 representing -18 dBm to +13 dBm.
                Default: 25 (+7 dBm)
  g<nnn>    - set Network Group  nnn - an integer (OEM default = 210)
  n<nn>     - set node ID n= an integer (standard node ids are 1..60)

  d<xx.x>   - xx.x = a floating point number for the datalogging period
  c<n>      - n = 0 for OFF, n = 1 for ON, enable voltage, current & power factor
                values to serial output for calibration.
  f<xx>     - xx = the line frequency in Hz: normally either 50 or 60
  k<x> <yy.y> <zz.z>
            - Calibrate an analogue input channel:
            - x = a single numeral: 0 = voltage calibration, 1 = ct1 calibration,
                  2 = ct2 calibration, etc
            - yy.y = a floating point number for the voltage/current calibration constant
            - zz.z = a floating point number for the phase calibration for this c.t.
                 (z is not needed, or ignored if supplied, when x = 0)
            -  e.g. k0 256.8
            -       k1 90.9 2.00
  a<xx.x>   - xx.x = a floating point number for the assumed voltage if no a.c.
                is detected
  m<x> <yy> - meter pulse counting:
               x = 0 for OFF, x = 1 for ON, <yy> = an integer for the pulse minimum
                period in ms. (y is not needed, or ignored when x = 0)
  t0 <y>    - turn temperature measurement on or off:
            - y = 0 for OFF, y = 1 for ON
  t<x> <yy> <yy> <yy> <yy> <yy> <yy> <yy> <yy>
            - change a temperature sensor's address or position:
            - x = a single numeral: the position of the sensor in the list (1-based)
            - yy = 8 hexadecimal bytes representing the sensor's address
               e.g.  28 81 43 31 07 00 00 D9
               N.B. Sensors CANNOT be added.
```

And the standard output (in whichever format you have chosen) will continue...

```
MSG:10,P1:31,P2:2224,P3:3829,E1:1,E2:61,E3:105,pulse:0
MSG:11,P1:35,P2:2236,P3:3850,E1:2,E2:67,E3:115,pulse:0
MSG:12,P1:34,P2:2234,P3:3847,E1:2,E2:73,E3:126,pulse:0
```

You can, and must, issue commands *between* the output messages, therefore you might find it convenient to set the datalogging period to a longer time, say 30 or 60 s, but you must remember to restore it before you save the settings and exit.

Exiting the configuration and calibration mode with **x** ensures that the access code will again be required, thereby reducing the risk of a change being made by accident.

If you exit *without* saving the settings, they will be used until the next restart, when it will revert to using the previous settings.

Note that Data Whitening is now done entirely within the RFM69CW, and must not be enabled in emonHub.

Using the emonTx with the ESP8266

The ESP8266 accepts serial data from the emonTx to send via WiFi. It also appears to send unwanted characters back to the emonTx, which conflict with the on-line calibration facility detailed above. It is for that reason that the **+++** access code is required. In order to

perform on-line calibration, the ESP8266 must be disconnected and a programmer connected to allow the user to interact with the emonTx via the Arduino IDE's monitor screen or equivalent. The emonTx must be powered and settings mode must be specifically enabled (issue the access code **+++**), and when complete and the settings saved, the emonTx can be powered down, when on-line calibration is disabled automatically. The ESP8266 can then be re-connected and both restarted.

Data Output

The output by radio is, in order:

- Message no.
- Vrms
- Powers 1-4
- Energies 1-4
- Temperatures 1-3
- Pulse Count.

The space-separated serial output is the same, but preceded by the NodeID.

The "key:value" pairs serial output for the EmonESP & EmonHubTx3eInterfacer sends the variables in the same order, but only the values in use are sent. The quantities are identified:

Message no.	MSG
Powers 1-4	P1 - P4
Energies 1-4	E1 - E4
Temperatures 1-3	T1 - T3
Pulse count	pulse